REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	November 5, 2007	Final Apr 15, 2006 – Jun 14, 2007

4. TITLE AND SUBTITLE
An Improved Testbed for Highly-Scalable Mission-Critical Information Systems

5. FUNDING NUMBERS
FA9550-06-1-0283

6. AUTHOR(S)
Van Renesse, Robbert

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Cornell University
120 Day Hall
Ithaca, NY 14853

8. PERFORMING ORGANIZATION REPORT NUMBER
49738

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR
4015 Wilson Boulevard Room 713
Arlington, VA 22203-1954

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
AFRL-SR-AR-TR-07-0524

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for Public Release; distribution is Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 Words)***

Network Emulation Testbeds are highly important resources to Computer Science research and to training students in the field of scalable network infrastructures. The Cornell DURIP cluster is anticipated to increase in usage over the next several years to include more members of the Systems groups here at Cornell University. We also anticipate a greater interaction with our colleagues at AFRL in Rome, NY. For this purpose we have upgraded the cluster so that it can run the Emulab software, to support 1Gbit networking, and to include 32 additional nodes.

14. SUBJECT TERMS
Scalable, Distributed, Gossip, Time-Critical, Event Delivery, Data Mining, Web Services

15. NUMBER OF PAGES
9

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
U	U	U	UU

FINAL PROGRESS REPORT

Network Emulation Testbeds are highly important resources to Computer Science research and to training students in the field of scalable network infrastructures. These are installations that are similar to compute clusters, but they are equipped with programmable network switches that allow various network topologies to be created. Network link characteristics such as bandwidth, latency, and message loss can be simulated in software (for example, using DummyNet). Compared to experiments run on live networks, results derived from a Network Emulation Testbed are reproducible and can be thoroughly understood through careful monitoring of the entire system

Contrast this with alternatives. While many CS departments have a large number of machines, it is impossible for students to reserve entire machines for some period of time, boot their own specialized operating systems, set up arbitrary network topologies, and shape network traffic. Similar restrictions also hold for PlanetLab, a large collection of nodes across the Internet that many CS departments have access to. Network Emulation Testbeds open the doors to a large class of research projects, and give students insight in the potentials and limitations of large interconnects.

Using a DURIP-03 grant, "A Testbed for Highly-Scalable Mission-Critical Information Systems," (Award Number F49620-03-1-0263) the Computer Science department at Cornell University has built a highly powerful 252-node testbed for experimenting with existing or novel pub/sub and other networking protocols used in mission-critical applications. In daily use still, it plays an important role in developing and evaluating new scalable fault-tolerant protocols and information systems.

For the current award, we proposed upgrading the cluster so that it can support Emulab. The reason for this was three-fold. First, the Emulab software would allow us to share the cluster resource much more effectively. Second, the Emulab software has a Web interface that allows people in remote locations (say, Rome Labs) to allocate and use the cluster resources without having to call in and get the help of a local person. Third, and perhaps most importantly, the Emulab software can program programmable switches to emulate complex network topologies

In the original proposal we requested a single used CISCO Catalyst switch with 100Mbit blades. After careful consideration and discussion with the Emulab people at Utah and the DETER/Emulab people at Berkeley we decided that it would behoove us to get Gigabit switches instead, and ended up purchasing 15 Nortel Baystack 5510 switches, each equipped with 48 1Gbit ports. These switches are also well supported by Emulab, support state-of-the-art network topologies, and the total price was actually less than the CISCO configuration.

The same people also strongly recommended not getting blades but 1U servers. So rather than 2 racks of blades for a total of 38 servers with 3 100Mbit ports each we ended up purchasing 32 1U servers with 3 1Gbit ports each, plus two larger 2U servers to support the Emulab software. Instead of 3 we were able to purchase 4 large storage servers to support experiments involving large distributed storage services, also requested in the original proposal.

The actual list of purchased equipment is then as follows:

1	Dell 24 U Short Rack
2	24 Outlet Switched Rack PDU
2	KVM w/ 16 Port Switch
2	NTS Super Micro 2U Server w/ Intel Pentium 4
3	Cyberpower powerstrip
4	Poweredge 2950 Rackmount Servers
6	6 Outlet Switched Rack PDU
15	Nortel Baystack 5510 48 Port
16	3 in 1 KVM Cables
17	Rail Kits
32	NTS Super Micro 1U Server w/ Intel Pentium 4
	Assorted Cat5 Cables

ONGOING ACTIVITIES INVOLVING THE USE OF THE CLUSTER

The old cluster is very heavily used and has led to many publications. We are excited to move to the new cluster because of its state-of-the-art capabilities and its ease-of-use, and potentially being able to federate with other clusters much adds to the research possibilities. We expect that many Cornell students will use or continue to the facility as the basis of research. In this connection, we expect that approximately 12-16 Cornell PhD students per year will perform experiments on the facility. The cluster also forms the basis for projects in Cornell's advanced graduate classes. Students engaged in such projects would potentially make use of DURIP equipment. This is expected to involve approximately 20 Cornell students per year, primarily from the MEng degree program. The facility, once federated, will also obtain workloads from Berkeley, ISI, and Vanderbilt. We will also make the cluster available to students from other places, on a case-by-case basis. We will now provide examples of some of the research projects that currently run at Cornell.

Scalable Services Architecture

Data centers constructed as clusters of inexpensive machines have compelling cost-performance benefits, but developing services to run on themcan be challenging. Our Scalable Services Architecture (SSA) helps developers build scalable clustered applications. The work is focused on nontransactional high-performance applications; these are poorly supported in existing platforms. A primary goal was to keep the SSA as small and simple as possible. Key elements include our TCP-based chain replication mechanism and a gossip-based subsystem for managing configuration data and repairing inconsistencies after faults.

Intrusion-Tolerant Live Streaming

Overlay networks provide routing functionality to applications without the need of having to change routers in the Internet. Overlay routing protocols are implemented entirely at the end hosts, and route messages from end host to end host. For example, a multicast routing protocol can be implemented by routing messages along a spanning tree of end hosts. Other functionality that may be supported this way includes resilient routing and content-based routing. Increasingly, mission-critical applications are built using overlay networks. But few of these overlay networks are tolerant of intrusions, making such networks fragile in the face of adversarial attacks. Application-level multicast systems are vulnerable to attacks that impede nodes from receiving desired data. Live streaming protocols are especially susceptible to packet loss induced by malicious behavior.

In this project we are working on an application-level live streaming system called SecureStream, built using a pull-based architecture that results in improved tolerance of malicious behavior. Our system is implemented as a layer running over our Fireflies system, an intrusion-tolerant membership protocol. Fireflies provides correct members with a view that includes all correct members and excludes crashed members. Fireflies exploits the robustness of epidemic protocols to defend against adversarial behavior.

Scalable Distributed Data Management

Peer-to-Peer (P2P) networks are emerging as a new paradigm for structuring large-scale distributed systems. The key advantages of P2P networks are their scalability, their fault tolerance, and their robustness, due to symmetrical nature of peers and self-organization in the face of failures. The above advantages made P2P networks suitable for content distribution and service discovery applications. However, many existing systems only support location of data items based on a key value (i.e. equality lookups). In many situations users will use their local servers to offer data or services described by semantically-rich XML documents. Users can then query this "P2P data warehouse" or "P2P service directory" as if all the data were stored in one huge centralized database.

As a first step towards this goal we have developed the P-tree, a new distributed fault-tolerant index structure that can efficiently support range queries in addition to equality queries. As an example, consider a large-scale computing grid distributed all over the world. Each grid node (peer) has an associated XML document that describes the node and its available resources. Specifically, each XML document has an IPAddress, an OSType, and a MainMemory attribute, each with the evident meaning. Given this setup, a user may wish to issue a query to find suitable peers for a main-memory intensive application. For example: "find peers with a Linux operating system with at least 4GB of main memory." The P-tree supports the above query efficiently as it supports both equality and range queries. In a stable system (no insertions or deletions), a P-tree of order d provides O(m + log dN) search cost for range

queries, where N is the number of peers in the system, m is the number of peers in the selected range and the cost is the number of messages. The P-tree requires O(d log dN) space at each peer and is resilient to failures of even large parts of the network. Our experimental results (both on a large-scale simulated network and in a small real network) show that P-trees can handle frequent insertions and deletions with low maintenance overhead and small impact on search performance.

Highly Scalable Distributed Data Stream Processing

Recently there has been considerable research on Data Stream Management Systems to support analysis of data that arrives rapidly in high-speed streams. Most of these systems have very expressive query languages in order to address a wide range of applications. We take a different approach. Instead of starting with a very powerful data stream query language, we begin with a well-known class of languages — event languages. Through the addition of several powerful language constructs (namely parameterization and aggregates), we extend their expressiveness towards full-fledged languages for processing data streams.

We have developed a novel algebra for expressing data stream queries, and a corresponding transformation of algebra expressions into finite state automata that can be implemented very efficiently. Our language is simple and natural, and it can express surprisingly powerful data stream queries like stateful subscriptions and parameterized sequence queries. To guarantee scalability for complex queries, we have developed techniques for effectively balancing load among several processors on a cluster. Our queries are similar to finite automata, and hence one approach is to distribute automata nodes and edges over several processors. The load on the system is also affected by the number of partially matched query patterns, for which incoming events are examined to determine if there are new matches. Distributing these partially matched patterns is another possibility for balancing load.

To study the tradeoffs of the different approaches in a real-world setting,
we intend to run the FingerLakes system on the DURIP facility. We will measure the effects of network bandwidth and latency, and we will study the tradeoff between communication overhead and effectiveness of load balancing for the different strategies. We have developed the query language and the theory behind the system. Furthermore, currently we have a prototype implementation of the single-processor Cayuga system. FingerLakes can re-use the Cayuga components and essentially adds a new layer on top for distributed stream processing. The current implementation makes use of QuickSilver multicast, discussed next.

QuickSilver Multicast (QSM)

In systems with large numbers of process groups, the individual groups mayheavily overlap. The overlap may not always be regular. Yet in many scenarios, such as when members include nodes of a cluster, or service replicas, regular overlap patterns might arise. Overlap hints to the possibility of sharing workload, yet most existing protocols do not benefit from it. In QSM we achieve scalability in the number of groups by exploiting the overlap to reduce the per-group overheads in both data dissemination and loss recovery.

How can overlap be used to improve performance? Lightweight group systems, such as Spread or the Isis Toolkit, map the application (lightweight) groups to a smaller set of heavyweight groups, multicast in the latter and filter on reception. Overlap permits batching. Messages destined to a large number of lightweight groups mapped to the same heavyweight group can be transmitted in a single packet. However, filtering incurs overhead. Also, such systems often rely on infrastructure nodes (agents) to relay messages, which increases latency, and in certain scenarios may lead to bottlenecks.

Can we benefit from across-group batching without the need for filtering or infrastructure nodes? In QSM, we achieve this by mapping groups to regions of overlap. A region is a set of nodes that are members of the same groups; formally, nodes x and y are in the same region iff $G(x) = G(y)$, where $G(x)$ is the set of all groups of which x is a member. Each node is normally in a single region. QSM employs a Global Membership Service (GMS) to process all group subscriptions, and to manage the group membership. The GMS determines the region boundaries and provides all members with consistent group and region membership notifications. To track membership, QSM uses a 2-level structure, in which both groups and regions are versioned.

We have done extensive experiments on our DURIP facility. We are able to stream close to 9000 1Kbyte message per second with 200 receivers. Thus, we are able to saturate the 100 MBit links. In order to observe scalability limits of our protocol, we need faster links and more nodes, as provided by the new cluster.

STATUS OF EFFORT

This hardware is now installed in its entirety and the Emulab software is installed and running. The old 252-node cluster is not yet integrated (it still runs experiments daily), but this integration will be done within the next several months.

PERSONNEL SUPPORTED

No personnel are supported under this grant. Various personnel who use the cluster themselves or have students using the cluster include

- Dr. Robbert van Renesse (Principal Investigator)
- Prof. Kenneth P. Birman
- Prof. Johannes Gehrke
- Prof. Fred B. Schneider
- Dr. Alan Demers
- Dr. Einar Vollset
- Dr. Hakim Weatherspoon

PUBLICATIONS

On the new cluster proper no publications have resulted as of yet. However, our research efforts make extensive use of the old DURIP facility and experiments will soon transition to make use of the upgrades. To see how successful these experiments are we list publications that used the DURIP resources for evaluation.

1. K. Ostrowski, K. Birman, D. Dolev. Live Distributed Objects: Enabling the Active Web. To Appear in IEEE Internet Computing.

2. Krzysztof Ostrowski, Ken Birman, and Danny Dolev. Extensible Architecture for High-Performance, Scalable, Reliable Publich-ubscribe Eventing and Notification. To Appear in the International Journal of Web Services Research. Volume 4, Number 4. October-December 2007.

3. T. Marian, M. Balakrishnan, K. Birman, R. van Renesse, and D. Dolev. Tempest: Scalable Time-Critical Web Services Platform. In Submission.

4. L. Ganesh, H. Weatherspoon, M. Balakrishnan and K. Birman. Optimizing Power Consumption in Large Scale Storage Systems. Proceedings of the 11th Workshop on Hot Topics in Operating Systems (HotOS XI). San Diego, CA. May 7-9, 2007.

5. K. Birman, A.-M. Kermarrec, K. Ostrowski, M. Bertier, D. Dolev, R. Van Renesse. Exploiting Gossip for Self-Management in Scalable Event Notification Systems. Distributed Event Processing Systems and Architecture Workshop (DEPSA). June 2007.

6. H. Johansen, D. Johansen, and R. van Renesse. *FirePatch: Secure and Time-Critical Dissemination of Software Patches.* IFIP International Information Security Conference (IFIPSEC 2007). Sandton, South-Africa. May 2007.

7. L. Brenna, A. Demers, J. Gehrke, M. Hong, J. Ossher, B. Panda, M. Riedewald, M. Thatte, and W. White. Cayuga: A High-Performance Event Processing Engine. In SIGMOD 2007.

8. A. Crainiceanu, P. Linga, A. Machanavajjhala, J. Gehrke, J. Shanmugasundaram. P-Ring: An Efficient and Robust P2P Range Index Structure. In SIGMOD 2007.

9. Mahesh Balakrishnan, Ken Birman, Amar Phanishayee, and Stefan Pleisch. Ricochet: Lateral Error Correction for Time-Critical Multicast. Proceedings of the 4th USENIX Symposium on Networked Systems Design & Implementation (NSDI 07). Cambridge, MA. April 2007.

10. K. Ostrowski, K. Birman, D. Dolev. Declarative Multi-Party Protocols. Cornell University Technical Report (TR2007-2088). April, 2007.

11. K. Ostrowski, K. Birman, D. Dolev. Implementing High-Performance Multicast in a Managed

Environment. Cornell University Technical Report (TR2007-2088). April, 2007.

12. K. Birman, M. Balakrishnan, D. Dolev, R. Marian, K. Ostrowski, A. Phanishayee. Scalable Multicast Platforms for a New Generation of Robust Distributed Applications. To Appear in Proceedings of the Second IEEE/Create-Net/ICST International Conference on Communication System software and Middleware (COMSWARE). Bangalore, India. January 7-12, 2007.

13. E. Vollset, K. Birman, and R. van Renesse. *Chickweed: Group Communication for Embedded Devices in Opportunistic Networking Environments*. 3rd International Workshop on Dependable Embedded Systems. Leeds, UK. October 2006.

14. M. Balakrishnan and K. Birman. Reliable multicast for time-critical systems. In Proceedings of the First IEEE Workshop on Applied Software Reliability (WASR 2006), Philadelphia, PA, June 2006.

15. M. Balakrishnan, K. Birman, and A. Phanishayee. PLATO: Predictive latency-aware total ordering. In Proceedings of the 25th IEEE Symposium on Reliable Distributed Systems, Leeds, UK, October 2006.

16. M. Balakrishnan, K. Birman, A. Phanishayee, and Stefan Pleisch. Ricochet: Low-latency multicast for scalable time-critical services. Technical report, Department of Computer Science, Cornell University, 2005.

17. K. Birman, M. Balakrishnan, D. Dolev, T. Marian, K. Ostrowski, and A. Phanishayee. Scalable multicast platforms for a new generation of robust distributed applications. In Proc. of the Second IEEE/CreateNet/ICST International Conference on Communication System software and Middleware (COMSWARE), Bangalore, India, January 2007.

18. K. Birman, R. van Renesse, and W. Vogels. Navigating in the storm: Using Astrolabe for distributed self-configuration, monitoring, and adaptation. Cluster Computing archive, 9(2), April 2006.

19. A. Bozdog, R. van Renesse, and D. Dumitriu. Selectcast – a scalable and self-repairing multicast overlay routing facility. In First ACM Workshop on Survivable and Self-Regenerative Systems, Fairfax, VA, October 2003.

20. I. Gupta, K. Birman, P. Linga, A. Demers, and R. van Renesse. Kelips: Building an efficient and stable P2P DHT through increased memory and background overhead. In International Workshop on Peer-to-Peer Systems (IPTPS '03), Berkeley, CA, February 2003.

21. M. Haridasan and R. van Renesse. Defense against intrusion in a live streaming multicast system. In 6th IEEE International Conference on Peer-to-Peer Computing (P2P2006), Cambridge, UK, September 2006.

22. M. Hicks, A. Nagarjan, and R. van Renesse. User-specified adaptive scheduling in a streaming media network. In OpenARCH'03, San Francisco, CA, April 2003.

23. H. Johansen, A. Allavena, and R. van Renesse. Fireflies: Scalable support for intrusion-tolerant network overlays. In Eurosys 2006, Leuven, Belgium, April 2006.

24. P. Linga, A. Crainiceanu, J. Gehrke, and J. Shanmugasundaram. Guaranteeing correctness and availabiliy in P2P range indices. In Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data (SIGMOD 2005), Baltimore, MD, June 2005.

25. P. Linga, I. Gupta, and K. Birman. Kache: Peer-to-peer web caching using Kelips. Submitted to ACM Transactions on Information Systems (TOIS), 2004.

26. T. Marian, K. Birman, and R. van Renesse. A scalable services architecture. In IEEE Symposium on Reliable Distributed Systems, Leeds, UK, October 2006.

27. K. Ostrowski and K. Birman. Extensible Web Services architecture for notification in large-scal systems. In Proceedings of the 2006 IEEE International Conference on Web Services (ICWS 2006), Chicago, IL September 2006.

28. K. Ostrowski and K. Birman. Scalable group communication system for scalable trust. In Proceedings of The First ACM Workshop on Scalable Trusted Computing (ACM STC 2006), Fairfax, VA, November 2006.

29. K. Ostrowski, K. Birman, and A. Phanishayee. The power of indirection: Achieving multicast scalability by mapping groups to regional underlays. Technical report, Department of Computer Science, Cornell University, November 2005.

30. R. van Renesse, K.P. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed systems monitoring, management, and data mining. ACM Transactions on Computer Systems, 21(3), May 2003.

31. R. van Renesse and F.B. Schneider. Chain replication for supporting high throughput and availability. In Sixth Symposium on Operating Systems Design and Implementation (OSDI '04), San Francisco, CA, December 2004.

INTERACTIONS/TRANSITIONS

Dr. Van Renesse and Prof. Birman are in active dialog with Prof. Anthony Joseph at Berkeley and with people at ISI, the University of Utah, and other places supporting clusters to investigate the possibility of federating our clusters. This would allow very large experiments to be conducted at virtually no extra costs.

Dr. Van Renesse has a dialog with ATC-NY (Dr. Mark Bickford and others) about a research project involving large MANET configurations and Byzantine defenses. The experiments for this research will be conducted on the new cluster.

Prof. Birman has a research effort with Dr.. A.-M. Kermarrec at INRIA involving the use of gossip in large scale multicast and pub/sub systems. INRIA (in France) has a cluster of its own, but we may consider experiments involving both clusters for additional scale).

Prof. Schneider, Prof. Gehrke, and Dr. Van Renesse do research on scalable fault-tolerant search engines in conjunction with researchers at the University of Tromso, Norway (Prof. Dag Johansen and Prof. Aage Kvalnes), and R&D staff at FAST Search and Transfer, a search engine company in Oslo. Schneider et al. intend to carry out simulation experiments on the new cluster.

Various other ongoing interactions may also lead to increased use of the cluster. For example, we have ongoing contacts with researchers at Rome Labs who have expressed interest in using the cluster.